Settings.js:  
// Settings.js

import React, {useState, useEffect, useCallback} from 'react';

import ClothingPreferences from './ClothingPreferences';

import {Link} from 'react-router-dom';

import {useAuth} from '../contexts/AuthContext';

function Settings() {

const {user}=useAuth();

const [preferences, setPreferences]=useState({

cold:'',

hot:'',

rainy:''

});

const [sensitivity, setSensitivity]=useState({

coldSensitive:false,

heatSensitive:false,

windSensitive:false

})

const [editMode, setEditMode]=useState(false);

const [message,setMessage]=useState('');

const [loading, setLoading]=useState(false);

//function to handle preference changes

const handleUpdatePreferences=(key, value)=>{

setPreferences(prev=>({...prev, [key]:value}))

}

//function to handle sensitivity changes

const handleUpdateSensitivity=(key, value)=>{

setSensitivity(prev=>({...prev, [key]:value}))

}

//function to handle form submission

const handleSubmitPreferences=async(e)=>{

e.preventDefault();

if(!user) return;

setLoading(true)

const payload={

\_id:user.\_id,

preferences,

sensitivity

}

console.log("Sending Update Payload:", payload)

try{

const response=await fetch(`http://localhost:5000/api/users/preferences`,{

method:'POST',

headers:{'Content-Type': 'application/json'},

body:JSON.stringify(payload)

})

if(!response.ok) throw new Error('Failed to update preferences')

setMessage('Preferences updated successfully!')

}catch(error){

console.error('Failed to submit preferences:', error)

setMessage(`Failed to update preferences: ${error.message}`)

}finally{

setLoading(false)

}

}

const[selectedCity, setSelectedCity]=useState('');

const canadaCities = [

{ name: "Toronto", lat: 43.651070, lon: -79.347015 },

{ name: "Vancouver", lat: 49.282730, lon: -123.120735 },

{ name: "Montreal", lat: 45.501690, lon: -73.567253 },

{ name: "Calgary", lat: 51.044734, lon: -114.071883 },

{ name: "Ottawa", lat: 45.421530, lon: -75.697193 },

{ name: "Edmonton", lat: 53.546124, lon: -113.493823 },

{ name: "Quebec City", lat: 46.813878, lon: -71.207981 },

{ name: "Winnipeg", lat: 49.895136, lon: -97.138374 },

{ name: "Halifax", lat: 44.648764, lon: -63.575239 },

{ name: "Victoria", lat: 48.428421, lon: -123.365644 },

{ name: "Saskatoon", lat: 52.157935, lon: -106.670906 },

{ name: "Regina", lat: 50.445211, lon: -104.618894 },

{ name: "St. John's", lat: 47.561510, lon: -52.712577 },

{ name: "Kelowna", lat: 49.888019, lon: -119.496011 },

{ name: "Barrie", lat: 44.389356, lon: -79.690332 },

{ name: "Sherbrooke", lat: 45.404171, lon: -71.892911 },

{ name: "Guelph", lat: 43.544805, lon: -80.248167 },

{ name: "Kingston", lat: 44.231172, lon: -76.485954 },

{ name: "Sudbury", lat: 46.491961, lon: -80.991211 },

{ name: "Thunder Bay", lat: 48.380895, lon: -89.247682 }

];

const fetchUserPreferences=useCallback(async()=>{

if(!user) return;

setLoading(true);

try{

const response=await fetch(`http://localhost:5000/api/users/preferences/${user.\_id}`)

const data=await response.json();

if(response.ok){

//debug

console.log("Fetched preferences:", data)

setPreferences(data.preferences || {cold:'', hot:'',rainy:''});

setSelectedCity(data.city||'Select a city')

}else{

throw new Error(data.message || 'Failed to fetch preferences')

}

}catch(error){

console.error('Error fetching preferences:',error)

setMessage(`Failed to fetch preferences: ${error.message}`)

}finally{

setLoading(false);

}

},[user]);

useEffect(()=>{

if(user){

fetchUserPreferences();

}

}, [user, fetchUserPreferences])

const handleLocationChange=(event)=>{

const city=canadaCities.find(city=>city.name===event.target.value)

setSelectedCity(event.target.value);

updateLocation(city);

}

const updateLocation=async(city)=>{

if(!user){

setMessage("User is not logged in/User data is not available.")

return;

}

setLoading(true);

try{

const response=await fetch('http://localhost:5000/api/users/update-city', {

method: 'POST',

headers:{'Content-Type':'application/json'},

body:JSON.stringify({

\_id:user.\_id,

city:city.name

})

})

if (!response.ok) {

const errorText = await response.text();

throw new Error(errorText || 'Failed to update location');

}

const data=await response.json();

setMessage(data.message);

setSelectedCity(city.name);

} catch (error) {

setMessage('Error updating location: ' + error.message);

} finally {

setLoading(false);

}

}

return (

<div>

<div className="setting-container">

<h1>Settings</h1>

{loading && <p>Loading...</p>}

{message&&<p>{message}</p>}

{user && (

<ClothingPreferences readOnly={!editMode} preferences={preferences} sensitivity={sensitivity}

updatePreferences={handleUpdatePreferences}

updateSensitivity={handleUpdateSensitivity}

handleSubmit={handleSubmitPreferences}/>

)}

{user&&(

<button onClick={() => setEditMode(!editMode)}>{editMode ? 'Save Changes' : 'Modify Preferences'}</button>

)}

<div>

<label htmlFor="city-select">Choose your city: </label>

<select id="city-select" value={selectedCity} onChange={handleLocationChange}>

<option value="">Select a city</option>

{canadaCities.map(city=>(

<option key={city.name} value={city.name}>{city.name}</option>

))}

</select>

</div>

</div>

<p>Back to <Link to="/">Main Page</Link></p>

</div>

);

}

export default Settings;

ClothingPreferences.js:  
  
import React from 'react';

function ClothingPreferences({preferences,sensitivity,updatePreferences, updateSensitivity, handleSubmit, readOnly}) {

const clothingOptions = {

cold: ["Heavy Coat", "Thermal Jacket", "Sweater", "Fleece Jacket", "Down Jacket", "Wool Coat", "Long Underwear", "Scarves, Hats, and Gloves", "Snow Boots", "Other"],

hot: ["T-shirt", "Shorts", "Light Dress","Tank Top", "Linen Shirt", "Sun Hat", "Sandals", "Capri Pants", "Other"],

rainy: ["Raincoat", "Waterproof Jacket", "Poncho", "Umbrella", "Waterproof Pants", "Rubber Boots", "Water-Resistant Backpack", "Waterproof Hat", "Quick-Dry Clothing", "Other"]

};

if(!preferences || !sensitivity){

return <p>Loading preferences...</p>

}

return (

<div>

<div className="preferences-container">

<h2>Set Your Preferences</h2>

<form onSubmit={handleSubmit}>

{Object.entries(clothingOptions).map(([key, items]) => (

<div key={key}>

<label>{`Preferred Clothing for ${key.charAt(0).toUpperCase() + key.slice(1)} Weather:`}</label>

<select name={key} value={preferences[key]} onChange={(e)=>updatePreferences(key, e.target.value)} disabled={readOnly}>

{items.map(item => (

<option key={item} value={item}>{item}</option>

))}

</select>

</div>

))}

<div>

<label>Cold Sensitive:</label>

<input type="checkbox" name="coldSensitive" checked={sensitivity.coldSensitive} onChange={(e)=>updateSensitivity('coldSensitive', e.target.checked)}disabled={readOnly} />

</div>

<div>

<label>Heat Sensitive:</label>

<input type="checkbox" name="heatSensitive" checked={sensitivity.heatSensitive} onChange={(e)=>updateSensitivity('heatSensitive', e.target.checked)} disabled={readOnly} />

</div>

<div>

<label>Wind Sensitive:</label>

<input type="checkbox" name="windSensitive" checked={sensitivity.windSensitive} onChange={(e)=>updateSensitivity('windSensitivity', e.target.checked)} disabled={readOnly} />

</div>

{!readOnly&&<button type="submit">Update Preferences</button>}

</form>

</div>

</div>

);

}

export default ClothingPreferences;

June 17, 2024

Settings.js:

// Settings.js

import React, {useState, useEffect, useCallback} from 'react';

import ClothingPreferences from './ClothingPreferences';

import {Link} from 'react-router-dom';

import {useAuth} from '../contexts/AuthContext';

function Settings() {

    const {user}=useAuth();

    const [preferences, setPreferences]=useState({

        cold:'',

        hot:'',

        rainy:''

    });

    const [sensitivity, setSensitivity]=useState({

        coldSensitive:false,

        heatSensitive:false,

        windSensitive:false

    })

    const [message,setMessage]=useState('');

    const [loading, setLoading]=useState(false);

    //function to update both preferences and city

    const handleSaveChanges=async()=>{

        if(!user) return;

        setLoading(true);

        try{

            const payload={

                \_id:user.\_id,

                preferences,

                sensitivity,

                city:selectedCity

            }

            console.log("Sending Update Payload:", payload)

            const response=await fetch(`http://localhost:5000/api/users/preferences`, {

                method:'POST',

                headers:{'Content-Type':'application/json'},

                body:JSON.stringify(payload)

            })

            if(!response.ok) throw new Error('Failed to update settings')

            setMessage('Settings updated successfully!')

        }catch(error){

            console.error('Failed to submit settings:', error);

            setMessage(`Failed to update settings: ${error.message}`)

        }finally{

            setLoading(false);

        }

    }

    const[selectedCity, setSelectedCity]=useState('');

    const canadaCities = [

        { name: "Toronto", lat: 43.651070, lon: -79.347015 },

        { name: "Vancouver", lat: 49.282730, lon: -123.120735 },

        { name: "Montreal", lat: 45.501690, lon: -73.567253 },

        { name: "Calgary", lat: 51.044734, lon: -114.071883 },

        { name: "Ottawa", lat: 45.421530, lon: -75.697193 },

        { name: "Edmonton", lat: 53.546124, lon: -113.493823 },

        { name: "Quebec City", lat: 46.813878, lon: -71.207981 },

        { name: "Winnipeg", lat: 49.895136, lon: -97.138374 },

        { name: "Halifax", lat: 44.648764, lon: -63.575239 },

        { name: "Victoria", lat: 48.428421, lon: -123.365644 },

        { name: "Saskatoon", lat: 52.157935, lon: -106.670906 },

        { name: "Regina", lat: 50.445211, lon: -104.618894 },

        { name: "St. John's", lat: 47.561510, lon: -52.712577 },

        { name: "Kelowna", lat: 49.888019, lon: -119.496011 },

        { name: "Barrie", lat: 44.389356, lon: -79.690332 },

        { name: "Sherbrooke", lat: 45.404171, lon: -71.892911 },

        { name: "Guelph", lat: 43.544805, lon: -80.248167 },

        { name: "Kingston", lat: 44.231172, lon: -76.485954 },

        { name: "Sudbury", lat: 46.491961, lon: -80.991211 },

        { name: "Thunder Bay", lat: 48.380895, lon: -89.247682 }

      ];

    const fetchUserPreferences=useCallback(async()=>{

        if(!user) return;

        setLoading(true);

        try{

            const response=await fetch(`http://localhost:5000/api/users/preferences/${user.\_id}`)

            const data=await response.json();

            if(response.ok){

                        //debug

             console.log("Fetched preferences:", data)

             setPreferences(data.preferences || {cold:'', hot:'',rainy:''});

             setSelectedCity(data.city||'Select a city')

            }else{

                throw new Error(data.message || 'Failed to fetch preferences')

            }

        }catch(error){

            console.error('Error fetching preferences:',error)

            setMessage(`Failed to fetch preferences: ${error.message}`)

        }finally{

            setLoading(false);

        }

    },[user]);

    useEffect(()=>{

        if(user){

            fetchUserPreferences();

        }

    }, [user, fetchUserPreferences])

    const handleLocationChange=(event)=>{

        const city=canadaCities.find(city=>city.name===event.target.value)

        setSelectedCity(event.target.value);

        updateLocation(city);

    }

    const updateLocation=async(city)=>{

        if(!user){

            setMessage("User is not logged in/User data is not available.")

            return;

        }

        setLoading(true);

        try{

            const response=await fetch('http://localhost:5000/api/users/update-city', {

                method: 'POST',

                headers:{'Content-Type':'application/json'},

                body:JSON.stringify({

                    \_id:user.\_id,

                    city:city.name

                })

            })

            if (!response.ok) {

                const errorText = await response.text();

                throw new Error(errorText || 'Failed to update location');

            }

            const data=await response.json();

            setMessage(data.message);

            setSelectedCity(city.name);

        } catch (error) {

            setMessage('Error updating location: ' + error.message);

        } finally {

            setLoading(false);

        }

    }

    return (

        <div>

            <div className="setting-container">

            <h1>Settings</h1>

            {loading && <p>Loading...</p>}

            {message&&<p>{message}</p>}

            {user && (

                <ClothingPreferences readOnly={false} preferences={preferences} sensitivity={sensitivity}

                updatePreferences={(key,value)=>setPreferences(prev=>({...prev,[key]:value}))}

                updateSensitivity={(key,value)=>setSensitivity(prev=>({...prev,[key]:value}))}

                />

            )}

            {user&&(

            <button onClick={handleSaveChanges}>Save Changes</button>

            )}

            <div>

                <label htmlFor="city-select">Choose your city: </label>

                <select id="city-select" value={selectedCity} onChange={handleLocationChange}>

                    <option value="">Select a city</option>

                    {canadaCities.map(city=>(

                        <option key={city.name} value={city.name}>{city.name}</option>

                    ))}

                </select>

            </div>

            </div>

            <p>Back to <Link to="/">Main Page</Link></p>

        </div>

    );

}

export default Settings;

ClothingPreferences.js:  
import React from 'react';

function ClothingPreferences({preferences,sensitivity,updatePreferences, updateSensitivity, handleSubmit, readOnly}) {

    const clothingOptions = {

        cold: ["Heavy Coat", "Thermal Jacket", "Sweater", "Fleece Jacket", "Down Jacket", "Wool Coat", "Long Underwear", "Scarves, Hats, and Gloves", "Snow Boots", "Other"],

        hot: ["T-shirt", "Shorts", "Light Dress","Tank Top", "Linen Shirt", "Sun Hat", "Sandals", "Capri Pants", "Other"],

        rainy: ["Raincoat", "Waterproof Jacket", "Poncho", "Umbrella", "Waterproof Pants", "Rubber Boots", "Water-Resistant Backpack", "Waterproof Hat", "Quick-Dry Clothing", "Other"]

    };

    if(!preferences || !sensitivity){

        return <p>Loading preferences...</p>

    }

    return (

        <div>

            <div className="preferences-container">

            <h2>Set Your Preferences</h2>

            <form onSubmit={handleSubmit}>

            {Object.entries(clothingOptions).map(([key, items]) => (

                    <div key={key}>

                        <label>{`Preferred Clothing for ${key.charAt(0).toUpperCase() + key.slice(1)} Weather:`}</label>

                        <select name={key} value={preferences[key]} onChange={(e)=>updatePreferences(key, e.target.value)} disabled={readOnly}>

                            {items.map(item => (

                                <option key={item} value={item}>{item}</option>

                            ))}

                        </select>

                    </div>

                ))}

                <div>

                    <label>Cold Sensitive:</label>

                    <input type="checkbox" name="coldSensitive" checked={sensitivity.coldSensitive} onChange={(e)=>updateSensitivity('coldSensitive', e.target.checked)}disabled={readOnly} />

                </div>

                <div>

                    <label>Heat Sensitive:</label>

                    <input type="checkbox" name="heatSensitive" checked={sensitivity.heatSensitive} onChange={(e)=>updateSensitivity('heatSensitive', e.target.checked)} disabled={readOnly} />

                </div>

                <div>

                    <label>Wind Sensitive:</label>

                    <input type="checkbox" name="windSensitive" checked={sensitivity.windSensitive} onChange={(e)=>updateSensitivity('windSensitivity', e.target.checked)} disabled={readOnly} />

                </div>

                {!readOnly&&<button type="submit">Update Preferences</button>}

            </form>

            </div>

        </div>

    );

}

export default ClothingPreferences;